UDACITY

PROJECT

Finding Donors for CharityML

A part of the Machine Learning Engineer Nanodegree Program

PROJECT REVIEW

NOTES

SHARE YOUR ACCOMPLISHMENT!

Meets Specifications

Dear student

Great start on this project! You've clearly understood the material from the tutorials and you've done a great job applying it to this real-world dataset. Congratulations on passing quickly and good luck with the next section of the course.

Cheers!

Exploring the Data

Student's implementation correctly calculates the following:

- **Number of records**
- **Number of individuals with income >$50,000**
- **Number of individuals with income <=$50,000**
- **Percentage of individuals with income > $50,000**

Perfect!

Preparing the Data

Student correctly implements one-hot encoding for the feature and income data.

Nice job! You can also use a lambda function to encode the labels too:

```
income = income_raw.apply(lambda x: 0 if x == '<=50K' else 1)
```

Evaluating Model Performance

Student correctly calculates the benchmark score of the naive predictor for both accuracy and F1 scores.

One interesting aspect to this predictor is that precision is equivalent to accuracy, and recall is always one. Hence a simpler implementation:

```
accuracy = n_greater_50k / n_records
fscore = (1.25) * accuracy / (0.25 * accuracy + 1)
```

The pros and cons or application for each model is provided with reasonable justification why each model was chosen to be explored.

Please list all the references you use while listing out your pros and cons.

Really nice work here! Here are some additional thoughts that I had:

> The main limitation of Random Forest is that a large number of trees can make the algorithm too slow and ineffective for real-time predictions. In general, these algorithms are fast to train, but quite slow to create predictions once they are trained. When run-time performance is more important, other approaches would be preferred.

This is definitely true. Another potential downside to this model is that it can be very difficult to interpret or understand how/why certain predictions are being made. Many people refer to Random Forests as a black box model. This isn't necessarily true as there are some mathematical hacks that allow a consensus tree to be visualized. However, this isn't at all straightforward or user friendly with most libraries that I'm familiar with.

> They don't perform so well in very large data sets because the training time happens to be cubic in the size of the data set. They also don't work well with lots of noise. When the classes are overlapping, independent evidence should be counted.

Another significant downside for this type of algorithm is that selecting a kernel that perfectly fits the dataset can be much more time intensive that most people realize. In some cases, a custom kernel function might even need to be coded after a significant statistical analysis.

Student successfully implements a pipeline in code that will train and predict on the supervised learning algorithm given.

The pipeline is perfectly implemented.

Student correctly implements three supervised learning models and produces a performance visualization.

```
clf_A = RandomForestClassifier(random_state=42)
clf_B = AdaBoostClassifier(random_state=42)
clf_C = SVC(random_state=42)
```

Nice job remembering to set random state values here!

Improving Results

Justification is provided for which model appears to be the best to use given computational cost, model performance, and the characteristics of the data.

Nice analysis here! Another thing that would be worth noting is that the Random Forests model is clearly overfitting (note the huge gap between the training and testing performance). There probably isn't a "bad" choice here, but I think you've made a good argument that AdaBoost is a good balance between optimal performance and computational cost.

Student is able to clearly and concisely describe how the optimal model works in layman's terms to someone who is not familiar with machine learning nor has a technical background.

> If this is still technical, Warren Smith uses a simple analogy in Quora to explain AdaBoost.

Nice link! I actually use this as an example for how to explain a model in layperson terminology. I think a metaphor is always a nice way to explain complex machine learning topics. The 'story' will help a non-technical person to retain at least a few of the details. Being able to do this well can be worth its weight in gold in industry. There are many times when we have to explain the 'big idea' to an employer or client in a way that makes sense

There are many times when we have to explain the "big idea" to an employer or client in a way that makes sense to them without seeming like we're talking down to them.

The final model chosen is correctly tuned using grid search with at least one parameter using at least three settings. If the model does not need any parameter tuning it is explicitly stated with reasonable justification.

Great work tuning your model!

Student reports the accuracy and F1 score of the optimized, unoptimized, models correctly in the table provided. Student compares the final model results to previous results obtained.

> They are better than the unoptimized model by a small factor.

Clearly the default parameter settings for this model are pretty good even without tuning.

Feature Importance

Student ranks five features which they believe to be the most relevant for predicting an individual's' income. Discussion is provided for why these features were chosen.

> sex - There is gender bias since males earn more than females for the same position.

It's unfortunate that this is probably true in many cases...

Student correctly implements a supervised learning model that makes use of the `feature_importances_` attribute. Additionally, student discusses the differences or similarities between the features they considered relevant and the reported relevant features.

> hours-per-week - Hours worked for full time and part time are different so more hours work means more income. Also, additional hours such as overtime may also increase income.

Another reason that this could be useful is that it can separate people with part time jobs from those with full time (salaried) positions. This could certainly help to drill down on the subset of people making >50K.

Keep in mind that these feature importances may be somewhat 'model specific'. In other words, if you re-ran the analysis with a DecisionTreeClassifier, you'd get a different list of 'most important' features. While it's

the analysis with a DecisionTreeClassifier, you'd get a different list of "most important" features. While it's important to know which features are helping the model the most, keep in mind that this may not necessarily indicate something that is fundamental about the dataset itself.

Student analyzes the final model's performance when only the top 5 features are used and compares this performance to the optimized model from Question 5.

DOWNLOAD PROJECT

RETURN TO PATH

Student FAQ